Amendments to the claims (this listing replaces all prior versions):

1. (Currently amended) A method for obtaining a cyclic redundancy code for a message, comprising:

storing a reciprocal-approximator for a generator polynomial in a storage;

separating the message into a plurality of segments;

~~moduloing~~ obtaining a remainder for each of the plurality of segments, including multiplying each segment by the stored reciprocal-approximator ~~a generator polynomial to obtain a remainder for each of the plurality of segments~~;

multiplying the remainder for each segment by a segment-constant based on the generator polynomial to obtain a plurality of segment-remainders;

accumulating the segment-remainders to obtain an accumulated-remainder; and

obtaining the cyclic redundancy code for the message, including multiplying ~~moduloing~~ the accumulated-remainder by the stored reciprocal-approximator ~~generator polynomial to obtain the cyclic redundancy code for the message~~.

2. (Cancelled)

3. (Original) The method of claim 1, further comprising separating the message into three or more segments.

4. (Original) The method of claim 1, wherein the cyclic redundancy code is appended to the message and the appended message is transmitted to a receiver.

5. (Original) The method of claim 1, wherein cyclic redundancy code indicates the existence of an error in the message.

6. (Original) The method of claim 5, wherein integrity of the message is verified if the cyclic redundancy code is zero.

7. (Previously Presented) The method of claim 5, wherein the integrity of the message is invalidated if the cyclic redundancy code is non-zero.

8. (Original) The method of claim 1, wherein moduloing includes dividing by the generator polynomial.

9. (Currently amended) The method of claim 1, wherein ~~moduloing includes multiplying by a~~ the reciprocal-approximator for the generator polynomial comprises X^{p+ra} / P, where P is the generator polynomial, p is the degree of the generator polynomial, and ra is the degree of the reciprocal-approximator.

10. (Original) The method of claim 1 wherein the segment-constant for each segment is obtained by moduloing the position of the segment in the message by the generator polynomial.

11. (Currently amended) A device for obtaining a cyclic redundancy code for a message, the message separated into a plurality of segments, comprising:

a storage to store a reciprocal-approximator for a generator polynomial;

a modulo unit to obtain a remainder for each of the plurality of segments by performing computations that include multiplying ~~modulo~~ each segment of the message by the stored reciprocal-approximator ~~a generator polynomial to obtain a remainder for each of the plurality of segments~~;

a multiplier to multiply the remainder for each segment by a segment-constant based on the generator polynomial to obtain a plurality of segment-remainders; and

an accumulator to accumulate the segment-remainders to obtain an accumulated-remainder;

wherein the modulo unit also obtains the cyclic redundancy code for the message by performing computations that include multiplying ~~modulos~~ the accumulated-remainder by the stored reciprocal-approximator ~~generator polynomial to obtain the cyclic redundancy code for the message~~.

12. (Previously Presented) The device in claim 11, wherein the device is a network card.

13. (Original) The device in claim 11, further comprising a memory for storing a plurality of segment-constants.

14. (Previously Presented) The device in claim 11, wherein the segment-constant is obtained upon receipt of the message.

15. (Original) The device in claim 11, wherein the modulo unit divides the accumulated-remainder by the generator polynomial to obtain the cyclic redundancy code.

16. (Currently amended) The device in claim 11, wherein the ~~modulo unit multiplies the accumulated-remainder by a~~ reciprocal-approximator for the generator polynomial comprises X^{p+ra} / P, where P is the generator polynomial, p is the degree of the generator polynomial, and ra is the degree of the reciprocal-approximator ~~to obtain the cyclic redundancy code~~.

17. (Original) A method for determining a cyclic redundancy code, comprising:

separating a message into a plurality of segments;

multiplying each segment by a segment-constant based on a generator polynomial to obtain a plurality of segment-remainders;

accumulating the segment-remainders to obtain an accumulated-remainder; and

moduloing the accumulated-remainder by the generator polynomial to obtain the cyclic redundancy code for the message.

18. (Original) The method of claim 17, where a degree of a most significant bit of the generator polynomial is greater than a degree of a most significant bit of each segment.

19. (Original) The method of claim 17, comprising separating the message into three or more segments.

20. (Previously Presented) The method of claim 17, wherein multiplying each segment by a segment-constant based on a generator polynomial (P) comprises multiplying each segment by a segment-constant based on a field extension F of the generator polynomial P, wherein F is equal to P multiplied by an extender Q.

21. (Original) The method of claim 17, wherein cyclic redundancy code indicates a likelihood of an error in the message.

22. (Original) The method of claim 17, wherein each one the plurality of segment-constants is based on the generator polynomial and the position of the segment in the message.

23. (Original) A device that obtains a cyclic redundancy code for a message, the message separated into a plurality of segments, comprising:

a multiplier to multiply each segment by a segment-constant to obtain a plurality of segment-remainders;

an accumulator to accumulate the segment-remainders to obtain an accumulated-remainder for the message; and

a modulo unit to modulo the accumulated-remainder by a generator polynomial to obtain the cyclic redundancy code for the message.

24. (Original) The device in claim 23, further comprising a memory for storing a plurality of segment-constants.

25. (Currently amended) The device in claim 23, wherein the modulo unit <u>modulos the accumulated-remainder by dividing</u> ~~divides~~ the accumulated-remainder by the generator polynomial to obtain the cyclic redundancy code.

26. (Original) The device in claim 23, wherein the modulo unit multiplies the accumulated-remainder by a reciprocal-approximator for the generator polynomial to obtain the cyclic redundancy code.

27-34. (Cancelled)

35. (Currently amended) An article comprising a machine-readable medium that stores instructions to obtain a cyclic redundancy code for a message, the instructions causing a machine to:

store a reciprocal-approximator for a generator polynomial in a storage;

separate the message into a plurality of segments;

~~modulo~~ obtain a remainder for each of the plurality of segments, including multiplying each segment by the stored reciprocal-approximator ~~a generator polynomial to obtain a remainder for each of the plurality of segments~~;

multiply the remainder for each segment by a segment-constant based on a generator polynomial to obtain a plurality of segment-remainders;

accumulate the segment-remainders to obtain an accumulated-remainder; and

~~modulo~~ obtain the cyclic redundancy code for the message, including multiplying the accumulated-remainder by the stored reciprocal-approximator ~~generator polynomial to obtain the cyclic redundancy code for the message~~.

36. (Original) The article of claim 35, further comprising instructions that cause a machine to modulo the segments by the generator polynomial to obtain the remainder for each segment.

37. (Original) The article of claim 35, further comprising instructions that cause a machine to verify the integrity of the message if the cyclic redundancy code is zero.

38. (Original) The article of claim 35, further comprising instructions that cause a machine to invalidate the integrity of the message if the cyclic redundancy code is non-zero.

39. (Original) An article comprising a machine-readable medium that stores instructions to obtain a cyclic redundancy code for a message, the instructions causing a machine to:

separate a message into a plurality of segments;

multiply each segment by a segment-constant based on a generator polynomial to obtain a plurality of segment-remainders;

accumulate the segment-remainders to obtain an accumulated-remainder; and

modulo the accumulated-remainder by the generator polynomial to obtain the cyclic redundancy code for the message.

40. (Original) The article of claim 39, further comprising instructions that cause a machine to apply a field extender to the generator polynomial.

41-45. (Cancelled)

46. (Previously Presented) The article of claim 39 in which the instructions causing the machine to multiply each segment by a segment-constant based on a generator polynomial (P) comprises instructions causing the machine to multiply each segment by a segment-constant based on a field extension F of the generator polynomial P, wherein F is equal to P multiplied by an extender Q.

47. (Previously Presented) The article of claim 46, wherein the greatest common denominator between P and Q is one.

48. (Previously Presented) The method of claim 20, wherein the greatest common denominator between P and Q is one.

49. (New) The article of claim 35, wherein the reciprocal-approximator for the generator polynomial comprises X^{p+ra} / P, where P is the generator polynomial, p is the degree of the generator polynomial, and ra is the degree of the reciprocal-approximator.

50. (New) A method for obtaining a cyclic redundancy code for a message, comprising:

separating the message into a plurality of segments;

obtaining a remainder for each of the plurality of segments, including multiplying each segment by a reciprocal-approximator that comprises X^{p+ra} / P, where P is a generator polynomial, p is the degree of the generator polynomial, and ra is the degree of the reciprocal-approximator;

multiplying the remainder for each segment by a segment-constant based on the generator polynomial to obtain a plurality of segment-remainders;

accumulating the segment-remainders to obtain an accumulated-remainder; and

obtain the cyclic redundancy code for the message, including multiplying the accumulated-remainder by the reciprocal-approximator.